User Manual

tRNAscan-SE: a program for improved transfer RNA detection in genomic sequence (release: 1.23)

Todd Lowe
School of Engineering
University of California
Santa Cruz, CA
lowe@soe.ucsc.edu

October 2001

1 Introduction

Note: An HTML version of this manual can be found on the web at "http://genome.wustl.edu/lowe/tRNAscan-SE-Manual/Manual.html".

1.1 Brief Description

tRNAscan-SE identifies transfer RNA genes in genomic DNA or RNA sequences. It combines the specificity of the Cove probabilistic RNA prediction package (Eddy & Durbin, 1994) with the speed and sensitivity of tRNAscan 1.3 (Fichant & Burks, 1991) plus an implementation of an algorithm described by Pavesi and colleagues (1994) which searches for eukaryotic pol III tRNA promoters (our implementation referred to as EufindtRNA). tRNAscan and EufindtRNA are used as first-pass prefilters to identify "candidate" tRNA regions of the sequence. These subsequences are then passed to Cove for further analysis, and output if Cove confirms the initial tRNA prediction. In this way, tRNAscan-SE attains the best of both worlds:

- a false positive rate of less than one per 15 billion nucleotides of random sequence
- the combined sensitivities of tRNAscan and EufindtRNA (detection of 99% of true tRNAs)
- search speed 1,000 to 3,000 times faster than Cove analysis and 30 to 90 times faster than the original tRNAscan 1.3 (tRNAscan-SE uses both a code-optimized version of tRNAscan 1.3 which gives a 650-fold increase in speed, and a fast C implementation of the Pavesi *et al.* algorithm).

This program and results of its analysis of a number of genomes have been published in Lowe & Eddy, *Nucleic Acids Research* **25**: 955-964 (1997).

1.2 What is included in this package?

This distribution includes the PERL script tRNAscan-SE, all the files necessary to compile and run the complete COVE package (version 2.4.4), all the files necessary to compile and run the modified version of tRNAscan (version 1.4), and all the files needed to compile and run EufindtRNA 1.0 (the cove programs, tRNAscan 1.4, and EufindtRNA are included for use with the tRNAscan-SE program, but may also be run as stand-alone programs). Installation of the PERL (Practical Extraction and Report Language, Larry Wall) interpreter package version 5.0 or later is required to run the tRNAscan-SE PERL script.

1.3 Getting Started

The following instructions for installing the tRNAscan-SE package also appear in the "INSTALL" text file:

1. Edit the top of the Makefile. Set the paths and other make variables to suit your system.

   In particular, you need to specify:

   - where executables are to be installed
   - where data files are to be installed
   - where Perl is already installed on the system
   - what the Perl executable is called (i.e. "perl" or "perl5")
   - where temporary files will reside
   - where to install man pages

2. type 'make' to build the programs

3. type 'make install' to install the programs and man pages

   Note: If you later manually move the location of binaries or data files from the directories specified in the 'Makefile', you need to delete the tRNAscan-SE executable, update the locations specified in the Makefile, and 'make install' again.

4. type 'make testrun' to run tRNAscan-SE on a sample sequence; if the program runs with no error messages, tRNAscan-SE has been installed correctly

5. type 'make clean' to clean up.

tRNAscan-SE is known to build cleanly on a number of different UNIX platforms and OS's, including SGI (IRIX), Sun (Solaris), DEC Alpha (OSF/1), and Intel x86 (Linux).

Note: A small Perl script (checkversion.pl) will run in the process of 'make'ing tRNAscan-SE. If you have specified an invalid location or version of Perl (<5.0) in the Makefile

PERLDIR variable, the make will fail. If you plan to install a correct version of Perl *after* installing the tRNAscan-SE package, you may short-circut this check by commenting out the line in the makefile containing 'checkversion.pl'.

Once installed, the user may wish to work through several of the examples included (Section 6 of this document) to get a quick feel for the program's operation and some of the most commonly used command line options. A description of the default run mode and output appears in Section 4, and a description of each of the program options is included in Section 5 of this document.

1.4 Intended Use

tRNAscan-SE was designed to make rapid, sensitive searches of genomic sequence feasible using the selectivity of the Cove analysis package. We have optimized search sensitivity with eukaryote cytoplasmic & eubacterial sequences, but it may be applied more broadly with a slight reduction in sensitivity.

1.5 Web Resources

For small-scale users and those who are unable to install tRNAscan-SE on a local UNIX platform, a web-based version of the program is available for on-line tRNA analysis at "http://genome.wustl.edu/eddy/tRNAscan-SE/". All of the most frequently used options are available in the web-based version. Links to the most recent release of the program and the tRNAscan-SE Genomic tRNA Database are also available from this page.

2 Methods

tRNAscan-SE does no tRNA detection itself, but instead combines the strengths of three independent tRNA prediction programs by negotiating the flow of information between them, performing a limited amount of post-processing, and outputting the results. The program works in three main phases. In the first stage, it runs two independent tRNA detection programs on the input DNA sequence. These relatively fast, first-pass detection programs include a modified, optimized version of tRNAscan 1.3 (1), and EufindtRNA, an implementation of another tRNA search algorithm previously described (3).

tRNAscan 1.3 detects tRNAs by initially looking for short, well conserved intragenic promoter sequences (A & B boxes in eukaryotes) found in the TPC and D arm regions of prototypic tRNAs. Once a specific number of nucleotides in the sequence match the consensus promoter (defined by an arbitrary score threshold), the program then progressively attempts to identify the various stem-loop structures found in the tRNA "clover leaf". As each arm is identified by the presence of base-pairing in the stem, correct loop size, and several invariant and semi-invariant bases, a "general score" counter is incremented. If the final score exceeds an empirically determined threshold, the tRNA location, anticodon, and type are saved.

EufindtRNA, on the other hand, only searches for linear sequence signals. A step-wise algorithm uses newly developed log-odds score matrices to first identify A and B box promoter elements that exceed an empirically determined cutoff. The scores for these A and B boxes are then added to a log odds score for the nucleotide distance between the A and B boxes to produce an intermediate score. Finally, a log odds score for the distance to the nearest downstream poly-T pol III termination signal is added to the intermediate score to obtain a final score. If the final score is above a final score cutoff, the tRNA identity and location is saved. tRNAscan-SE uses a less selective version of this algorithm that does not look for pol III termination signals, thus uses the intermediate score as a final cutoff. Also, the intermediate score cutoff is loosened slightly relative to the intermediate cutoff described in the original algorithm (3). These modifications increase the algorithm's sensitivity but greatly reduce EufindtRNA's selectivity. This does not reduce the final selectivity of tRNAscan-SE since a secondary filter (Cove) is being used to eliminate false positives. The sensitivity of EufindtRNA is roughly comparable to tRNAscan 1.3, but it appears to be complementary in that EufindtRNA tends to identify tRNAs missed by tRNAscan 1.3 and vice versa (3). tRNAscan-SE takes advantage of this fact, and saves results from both tRNAscan 1.3 and EufindtRNA, then merges them into one list of non-redundant "candidate" tRNA identifications.

In the second stage, tRNAscan-SE extracts the DNA subsequences identified as possible tRNAs and passes only these segments to an RNA search program in the Cove program suite (covels) for analysis. Cove programs look for tRNAs in a very different way. A probabilistic model for tRNA has been developed by aligning known tRNAs and giving a base-specific probability score to every nucleotide in the tRNA model. Also, Cove uses a special method for capturing secondary RNA structure information using a type of language referred to as a stochastic context-free grammar (SCFG). Cove applies this probabilistic model to the entire windowed sequence, and produces a probability score that the sequence matches the tRNA model. If the score exceeds 20.0 bits, the tRNA is considered a true tRNA (based on empirical studies in ref. 2).

In the final phase, tRNAscan-SE takes those tRNAs confirmed as such and runs another Cove program (coves) that displays RNA secondary structure. The tRNA type is predicted by identifying the anticodon within the structure output. Introns are also automatically identified from the structure output as runs of five or more consecutive non-consensus nucleotides within the anticodon loop.

tRNAscan-SE uses heuristics to try to distinguish pseudogenes from true tRNAs, primarily on lack of tRNA-like secondary structure. A second tRNA covariance model was created from the original 1415-tRNA alignment, under the constraint that no secondary structure is conserved (this model is effectively just a sequence profile, or hidden Markov model). By subtracting a tRNA's similarity score to the primary structure-only model from that using the complete tRNA model, a secondary structure-only score is obtained. We have observed that tRNAs with low scores for either component of the total score were often pseudogenes. Thus, tRNAs are marked as likely pseudogenes if they have either a score of less than 10 bits for the primary sequence component of the total score, or a score of less than 5 bits for the secondary structure component of the total score. Selenocysteine tRNAs are not checked by these rules since they have atypical primary and secondary structure. Also, use of the -O option (search for organellar tRNAs) disables pseudogene checking since these criteria are geared towards detecting cytoplasmic pseudogenes (some true non-eukaryotic tRNA are marked as pseudogenes by this analysis). Final tRNA predictions are then saved in tabular, ACeDB, or secondary structure output format.

For more details on the program algorithm & implementation, see the Nucleic Acids Research paper (Lowe & Eddy, 1997).

3 Performance / Requirements

Performance will obviously vary depending on the machine architecture, memory, and OS efficiency. The examples included in this document were run on a Silicon Graphics Indigo2 R4400-200 running IRIX 5.3, with over 32Mb of memory. tRNAscan-SE runs at approximately 20,000 to 45,000 bp/sec in its default operation mode on this machine.

4 Default Program Operation

4.1 Invoking tRNAscan-SE

The program is invoked by giving it a series of optional command line parameters, then a list of one or more sequence files written in the FASTA format (see appendix A for example of FASTA format):

```
tRNAscan-SE [-options] FASTA_file(s)
```

By default, the header credits and selected command-line options are printed to the screen via standard error, followed by the final results of the tRNA search written to standard output in a tabular format (see below). By default, tRNAscan-SE searches for eukaryotic cytoplasmic tRNAs. To search for prokaryotic, archaeal, or organellar tRNAs, use search mode options -P, -A, -O, repectively. If the sequences are from more than one phylogenetic domain, the general tRNA model (option -G) may be used with minimal loss of sensitivity and selectivity (the publication describing tRNAscan-SE used the general tRNA model exclusively, ref. 4).

```
Sequence              tRNA Bounds      tRNA   Anti    Intron Bounds   Cove
Name          tRNA #  Begin    End     Type   Codon   Begin    End    Score
--------      ------  -----    ---     ----   -----   -----    -----  -----
CELF22B7      1       12619    12738   Leu    CAA     12657    12692  60.01
CELF22B7      2       19480    19561   Ser    AGA     0        0      80.44
CELF22B7      3       26367    26439   Phe    GAA     0        0      80.32
CELF22B7      4       26992    26920   Phe    GAA     0        0      80.32
CELF22B7      5       23765    23694   Pro    CGG     0        0      75.76
```

Each new tRNA in a sequence is consecutively numbered in the 'tRNA #' column. 'tRNA Bounds' specify the starting (5') and ending (3') nucleotide bounds for the tRNA. tRNAs found on the reverse (lower) strand are indicated by having the Begin (5') bound greater than the End (3') bound (see tRNAs #4 & #5 in output above).

The 'tRNA Type' is the predicted amino acid charged to the tRNA molecule based on the predicted 'Anticodon' (written 5' $\rightarrow$ 3') displayed in the next column. tRNAs that fit criteria for potential pseudogenes (poor primary or secondary structure, discussed in Methods), will be marked with "Pseudo" in the 'tRNA Type' column. If there is a predicted intron in the tRNA, the next two columns indicate the nucleotide bounds. If there is no predicted intron, both of these columns contain zero. The final column is the Cove score for the tRNA in bits. Note that this score will vary somewhat depending on the particular tRNA covariance model used in the analysis (the search mode selects which tRNA covariance model will be used: eukaryote-specific, prokaryote-specific, archae-specific, or general). tRNAscan-SE counts any sequence that attains a score of $\geq$ 20.0 bits as a tRNA (based on empirical studies conducted by Eddy & Durbin in ref #2).

4.2 Temporary files

In the course of program execution, several temporary files are written to and deleted from the 'TEMPDIR' directory specified in the Makefile on installing the program. Alternatively, the environment variable 'TMPDIR' can be set to another directory which will override the temporary directory specified in the Makefile.

For the average user, /tmp should work fine as the temp file directory. For sequencing centers or users scanning very large individual sequences ($\geq$ 1MBp), or many sequences at once ($\geq$ 4 instances of tRNAscan-SE at once), it might be advisable to use /usr/tmp or some other temporary directory that has sufficient free disk space (at least 10MB free).

Note: If multiple FASTA files are specified on the command line, tRNAscan-SE creates a temporary file `tscan`*`process-id-number`*`.mseq` in which all sequence files are concatenated together for ease of processing. Because of this, the temporary directory must have enough room to temporarily save a copy of all the sequence files (at once) that have been specified on the command line — this may be a problem for "power users" who may conceivably scan an entire directory of cosmids totalling many MBp of sequence. In these cases, I would advise the user to either run a smaller set of sequences at once, or make sure the TEMPDIR can handle the large '.mseq' temporary file.

5 Command-line Options

5.1 Search Mode Options

By default, the eukaryotic tRNA model is used for tRNA analysis. To select an alternate tRNA model for sequences from other sources (other phylogenetic domains or mitochondria/chloroplasts), use one of the following options:

-B : search for bacterial tRNAs

This option selects the bacterial covariace model for tRNA analysis, and loosens the search parameters for EufindtRNA to improve detection of bacterial tRNAs. Use of this mode with bacterial sequences will also improve bounds prediction of the 3' end (the terminal CAA triplet).

-A : search for archaeal tRNAs

This option selects an archaeal-specific covariance model for tRNA analysis, as well as slightly loosening the EufindtRNA search cutoffs.

-O : search for organellar (mitochondrial/chloroplast) tRNAs

This parameter bypasses the fast first-pass scanners that are poor at detecting organellar tRNAs and runs Cove analysis only. Since true organellar tRNAs have been found to have Cove scores between 15 and 20 bits, the search cutoff is lowered from 20 to 15 bits. Also, pseudogene checking is disabled since it is only applicable to eukaryotic cytoplasmic tRNA pseudogenes. Since Cove-only mode is used, searches will be very slow (see -C option below) relative to the default mode.

-G : use general tRNA model

This option selects the original tRNA covariance model that was trained on tRNAs from all three phylogenetic domains (archaea, bacteria, & eukarya). This mode can be used when analyzing a mixed collection of sequences from more than one phylogenetic domain, with only slight loss of sensitivity and selectivity. The original publication describing this program and tRNAscan-SE version 1.0 used this general tRNA model exclusively. If you wish to compare scores to those found in the paper or scans using v1.0, use this option. Use of this option is compatible with all other search mode options described in this section.

-C : search using Cove analysis only

Directs tRNAscan-SE to analyze sequences using Cove analysis only. This option allows a slightly more sensitive search than the default tRNAscan + EufindtRNA ⇒ Cove mode, but is much slower (by approx. 250 to 3,000 fold). Output format and other program defaults are otherwise identical to the normal analysis.

-H : show both primary & secondary structure score components to covariance model bit scores

This option displays the breakdown of the two components of the covariance model bit score. Since tRNA pseudogenes often have one very low component (good secondary structure but poor primary sequence similarity to the tRNA model, or vice versa), this information may be useful in deciding whether a low-scoring tRNA is likely to be a pseudogene. The heuristic pseudogene detection filter uses this information to flag possible pseudogenes – use this option to see why a hit is marked as a possible pseudogene. It may be helpful to examine score breakdowns from known tRNAs in the organism of interest to get a frame of reference.

-D : disable pseudogene checking

Manually disable checking tRNAs for poor primary or secondary structure scores often indicative of eukaryotic pseudogenes. This will slightly speed the program & may be necessary for non-eukaryotic sequences that are flagged as possible pseudogenes but are known to be functional tRNAs.

5.2 Output Options

-o *file* : save final results in *file*

Specifiy this option to write results to *file* rather than standard output.

-f *file* : save results and Cove tRNA secondary structures to *file*

This option saves results and secondary structure information (as predicted by the coves program) in *file*. Use "$" in place of *file* to send to standard output. An example of the output format for one tRNA appears below:

```
CELF22B7.trna4 (26992-26920)    Length: 73 bp
Type: Phe       Anticodon: GAA at 34-36 (26959-26957)   Score: 73.88
         *    |    *    |    *    |    *    |    *    |    *    |    *    |
Seq: GCCTCGATAGCTCAGTTGGGAGAGCGTACGACTGAAGATCGTAAGGtCACCAGTTCGATCCTGGTTCGGGGCA
Str: >>>>>>>..>>>>........<<<<.>>>>>.......<<<<<.....>>>>>.......<<<<<<<<<<<<.

     |     |  |              | |               |     |               ||     |
     +-----+  +--------------+ +---------------+     +---------------++-----+
        |        D-stem/loop       Anticodon            TPC stem/loop    |
        |                           stem/loop                            |
        +----------------------------------------------------------------+
                                Isoacceptor stem
```

The first line contains the sequence name, trna#, tRNA bounds (in parentheses), and length of the tRNA. The next line contains the isoacceptor tRNA Type, Anticodon (with tRNA-relative and sequence-absolute bounds), and the Cove Score. This is identical information as would be seen in the tabular output format, excluding the anticodon bounds. The next line contains hash marks every 5 and 10 bp to ease position identification in the tRNA sequence that appears on the following line. On the sequence line, nucleotides matching the "consensus" tRNA model used in Cove analysis appear in upper case, while introns and other nucleotides in non-conserved positions are printed in lower-case letters. The last line contains predicted secondary structure folding of the tRNA, with nested '>' and '<' symbols representing base pairings. The various tRNA features are labelled in this example.

-a : output results in ACeDB output format

This option allows results to be written in ACeDB format instead of the default tabular output format.

-m *file* : save statistics summary for run

This option directs tRNAscan-SE to write a brief summary to *file* which contains the run options selected as well as statistics on the number of tRNAs detected at each phase of the search, search speed, and other statistics (examples on following pages).

Following is a description of each of these statistics, followed by an example stats summary file created from scanning the C. elegans cosmid F59C12:

```
tRNAscan-SE run results (on host <computer name>)
Started: <time & date tRNAscan-SE began>

<Parameters used for search printed here>

First-pass (tRNAscan/EufindtRNA) Stats:
---------------------------------------
Sequences read:         <total # of FASTA sequences read>
Seqs w/at least 1 hit:  <total sequences with at least one tRNA predicted>
Bases read:             <total nucleotides in all sequences searched (both strands)>
Bases in tRNAs:         <total nucleotides in tRNAs predicted>
tRNAs predicted:        <# tRNAs predicted from first-pass search program(s)>
Av. tRNA length:        <average tRNA length>
Script CPU time:        <CPU time spent by tRNAscan-SE
                         reading seqs, setting up run & writing results>
Scan CPU time:          <CPU time spent by tRNAscan/EufindtRNA finding tRNAs>
Scan speed:             <Averaged tRNAscan+EufindtRNA search speed>

First pass search(es) ended: <time & date tRNAscan/EufindtRNA searches
                              finished, Cove analysis begins>

Cove Stats:
-----------
Candidate tRNAs read:    <number of tRNAs detected by tRNAscan/EufindtRNA
                          that were passed to Cove for verification>
Cove-confirmed tRNAs:    <number of tRNAs positively confirmed by Cove>
Bases scanned by covels: <total nucleotides in all tRNAs searched by
                          Cove (covels-SE program) analysis>
% seq scanned by covels: <percent of total nucleotides in input
                          sequences that were analyzed by Cove>
Script CPU time:         <CPU time spent by tRNAscan-SE reading
                          seqs, setting up runs & writing results>
Cove CPU time:           <CPU time spent by Cove analysis programs>
Scan speed:              <Average Cove search speed>

Cove analysis of tRNAs ended: <time & date Cove analysis
                               and tRNAscan-SE completed>
Summary
--------
Confirmed tRNAs:     <total tRNAs predicted by tRNAscan-SE>
Overall scan speed:  <Average search speed for tRNAscan-SE>
```

An example stats summary file using the default search options on cosmid F59C12 follows:

```
tRNAscan-SE run results (on host wol)
Started: Wed Feb  5 15:02:30 CST 1997

------------------------------------------------------------
Sequence file(s) to search:  F59C12.fa
Results written to:          Standard output
Output format:               Tabular
Searching with:              tRNAscan + EufindtRNA -> Cove
tRNAscan parameters:         Strict
EufindtRNA parameters:       Relaxed (Int Cutoff= -32.1)
Search statistics saved in:  F59C12.stats
------------------------------------------------------------

First-pass (tRNAscan/EufindtRNA) Stats:
---------------
Sequences read:         1
Seqs w/at least 1 hit:  1
Bases read:             57976 (both strands)
Bases in tRNAs:         299
tRNAs predicted:        4
Av. tRNA length:        74
Script CPU time:        0.19 s
Scan CPU time:          0.55 s
Scan speed:             105.4 Kbp/sec

First pass search(es) ended: Wed Feb  5 15:02:31 CST 1997

Cove Stats:
-----------
Candidate tRNAs read:     4
Cove-confirmed tRNAs:     3
Bases scanned by covels:  355
% seq scanned by covels:  0.6 %
Script CPU time:          0.24 s
Cove CPU time:            8.08 s
Scan speed:               43.9 bp/sec

Cove analysis of tRNAs ended: Wed Feb  5 15:02:43 CST 1997

Summary
--------
Confirmed tRNAs:    3
Overall scan speed: 6399.1 bp/sec
```

-d : display program progress messages

This option directs the program to print messages indicating the progress of the program to standard output. If final results are also being sent to standard output, some of these messages will be suppressed so as to not interrupt display of the results.

-l *file* : save log of program progress in *file*

Identical to -d option, but sends message to *file* instead of standard output.
Note: the -d option overrides the -l option if both are specified on the same command line.

-q : quiet mode (credits & run option selections suppressed)

This option suppresses the program credits and run option selections normally printed to standard error at the beginning of each run.

-b : brief output format (no column headers)

This option eliminates column headers that appear by default when writing results in tabular output format. Useful if results are to be parsed or sent to another program.

-N : output corresponding codons instead of tRNA anticodons

This option causes tRNAscan-SE to output a tRNA's corresponding codon in place of its anticodon.

-? # : use of '#' symbol in specifying output file names

The '#' symbol may be used as shorthand to specify "default" file names for output files. The default file names are constructed by using the input sequence file name, followed by an extension specifying the output file type *seqfile.ext* where *.ext* is:

Extension	Produced by Option	Description
.out	-o	final output results (tabular or ACeDB format)
.stats	-m	summary statistics file
.log	-l	run progress file
.ss	-f	secondary structures save file
.fpass.out	-r	formatted, tabular output from first-pass runs
.fpos	-F	FASTA file of tRNAs identified in first-pass scans that were found to be false positives by Cove analysis

Note:

- if the input sequence file name has the extensions '.fa' or '.seq', these extensions will be removed before using the filename as a prefix for default file names. (example: input file name 'Mygene.seq' will have the output file name 'Mygene.out' if '#' is used with the -o option).

– if more than one sequence file is specified on the command line, the "default" output file prefix will be the name of the FIRST sequence file on the command line. Use the -p option (described next) to change this default name to something more appropriate when specifying more than one sequence file on the command line.

-p *label* : use *label* prefix for all output files

This option allows the user to specify the default output file prefix when using the '#' file name specification (instead of using the input sequence file name).

-y : show origin of first-pass hits

This option displays which of the first-pass scanners detected the tRNA being output. "Ts", "Eu", or "Bo" will appear in the last column of Tabular output, indicating that either tRNAscan 1.4, EufindtRNA, or both scanners detected the tRNA, respectively.

5.3 Specify Alternate Cutoffs / Data Files

-X *score* : set Cove cutoff score for reporting tRNAs (default=20)

This option allows the user to specify a different Cove score threshold for reporting tRNAs. It is not recommended that novice users change this cutoff, as a lower cutoff score will increase the number of pseudogenes and other false positives found by tRNAscan-SE (especially when used with the "Cove only" scan mode). Conversely, a higher cutoff than 20.0 bits will likely cause true tRNAs to be missed (numerous "real" tRNAs have been found just above the 20.0 cutoff). Knowledgable users may wish to experiment with this parameter to find unusual tRNAs or pseudogenes beyond the normal range of detection, keeping the preceding caveats in mind.

-L *length* : set max length of tRNA intron+variable region (default=116bp)

The default maximum tRNA length for tRNAscan-SE is 192 bp, but this limit can be increased with this option to allow searches with no practical limit on tRNA length. In the first phase of tRNAscan-SE, EufindtRNA searches for A and B boxes of ¡length¿ maximum distance apart, and passes only the 5' and 3' tRNA ends to covariance model analysis for confirmation (removing the bulk of long intervening sequences). tRNAs containing group I and II introns have been detected by setting this parameter to over 800 bp. Caution: group I or II introns in tRNAs tend to occur in positions other than the canonical position of protein-spliced introns, so tRNAscan-SE mispredicts the intron bounds and anticodon sequence for these cases. tRNA bound predictions, however, have been found to be reliable in these same tRNAs.

-I *score* : manually set "intermediate" cutoff score for EufindtRNA

This score cutoff affects the sensitivity of the first-pass scanner EufindtRNA. This parameter should not need to be adjusted from its default values (variable depending on search mode), but is included for users who are familiar with the Pavesi *et al.* (1994) paper and wish to set it manually. See Lowe & Eddy (1997) for details on parameter values used by tRNAscan-SE depending on the search mode.

-z *number* : use *number* nucleotides padding for first-pass tRNA predictions

By default, tRNAscan-SE adds 7 nucleotides to both ends of tRNA predictions when first-pass tRNA predictions are passed to covariance model (CM) analysis. CM analysis generally trims these bounds back down, but on occassion, allows prediction of an otherwise truncated first-pass tRNA prediction.

-g *file* : use alternate genetic codes specified in *file* for determining tRNA type

By default, tRNAscan-SE uses a standard universal codon → amino acid translation table that is specified at the end of the tRNAscan-SE.src source file. In many mitochondrial and a number of other microbial organisms, there are exceptions to this universal translation code. This option allows the user to specify exceptions to the universal code. Several alternate translation code files are included in this package for convenience:

```
gcode.cilnuc for Ciliates, Dasycladacean, & Hexamita nuclear tRNAs
gcode.echdmito for Echinoderm mitochondrial tRNAs
gcode.invmito for Invertibrate mitochondrial tRNAs
gcode.othmito for Mold, Protozoans, & Coelenterate mitochondrial tRNAs
gcode.vertmito for Vertibrate mitochondrial tRNAs
gcode.ystmito  for Yeast mitochondrial tRNAs
```

The user may also create a new alternate translation file. An example of an alternate translation code specification file follows:

```
# Vertebrate mitochondrial translation codes
# Format:  <Codon> <3-letter AA abbreviation> <One letter AA abrev>

TGA     Trp W
ATA     Met M
AGR     Stp *
```

Comments or other information will be ignored on lines preceded by a pound symbol '#'. Anticodon translation codes are specified by placing the three base codon, the three letter amino acid abbreviation, and the single letter amino acid abbreviation all on a single line (each separated by a space or tab). Degenerate symbols such as 'N', 'R', and 'Y' may also be used for codon specification. Any codon not specified in the alternate genetic code file will use the translation in the default 'universal' genetic code table that occurs at the very end of the tRNAscan-SE.src source code file. Changes can be made directly to the default translation table, but a new 'make install' must be run to install the modified PERL script.
Note: this option does not have any effect when using the -T or -E option (search using tRNAscan or EufindtRNA only) — you must be running in default or Cove only analysis mode.

-c *file* : use an alternate covariance model specified in *file*

For users who have developed their own tRNA covariance models using the Cove program "coveb" (see Cove documentation), this parameter allows substitution for the default tRNA covariance models. May be useful for extending Cove-only mode detection of particularly strange tRNA species such as mitochondrial tRNAs.

5.4 Miscellaneous Options

-h : print full list of available program options

Prints this list of program options, each with a brief, one-line description.

-Q : do not prompt user before overwriting pre-existing files

By default, if an output result file to be written to already exists, the user is prompted whether the file should be over-written or appended to. Using this options forces overwriting of pre-existing files without an interactive prompt. This option may be handy for batch-processing and running tRNAscan-SE in the background.

-n *expr* : search only sequences with names matching *expr* string

This option allows analysis of selected sequences in a sequence file containing multiple sequences. Only those sequences with names (first non-white space word after ">" symbol on FASTA name/description line) matching *expr* are analyzed. *expr* may contain * or ? wildcard characters, but the user should enclose such expressions in single quotes (for example: -n 'HU?alpha*') to prevent the shell from attempting to expand wildcards into file name matches.

-s *expr* : start search at sequence with name matching *expr* string and continue to end of input sequence file(s)

This option directs the program to analyze the first sequence with a name matching *expr*, and every sequence thereafter. This may be useful for re-starting crashed/aborted runs at the point where the previous run stopped. (if same names for output file(s) are used, program will ask if files should be over-written or appended to — choose append and run will successfully be restarted where it left off).

5.5 Options for testing & special applications

-T : search using tRNAscan only (defaults to strict search params)

Directs tRNAscan-SE to use only tRNAscan to analyze sequences. This mode will cause tRNAscan to default to using "strict" parameters (similar to tRNAscan version 1.3 operation). This mode of operation is slightly faster (about 3-5 times faster than default mode analysis), but will result in approximately 0.2 to 0.6 false positive tRNAs per Mbp, decreased sensitivity, and less reliable prediction of anticodons, tRNA isotype, and introns.

-t *mode* : explicitly set tRNAscan params, where *mode* = R or S (R=relaxed, S=strict tRNAscan v1.3 params)

This option allows selection of strict or relaxed search parameters for tRNAscan analysis. By default, "strict" parameters are used. Relaxed parameters may give very slightly increased search sensitivity, but increase search time by 20-40 fold.

-E : search using Eukaryotic tRNA finder (EufindtRNA) only

This option runs EufindtRNA alone to search for tRNAs. Since Cove is not being used as a secondary filter to remove false positives, this run mode defaults to "normal" parameters which more closely approximates the sensitivity and selectivity of the original algorithm describe by Pavesi and colleagues (see the next option, -e for a description of the various run modes).

-e *mode* : explicitly set EufindtRNA params, where *mode* = R, N, or S (relaxed, normal, or strict)

This option allows the user to explicitly set the parameters for EufindtRNA. The "relaxed" mode is used for EufindtRNA when using tRNAscan-SE in default mode. With relaxed parameters, tRNAs that lack pol III poly-T terminators are not penalized, increasing search sensitivity, but decreasing selectivity. When Cove analysis is being used as a secondary filter for false positives (as in tRNAscan-SE's default mode), overall selectivity is not decreased.

Using "normal" parameters with EufindtRNA does incorporate a log odds score for the distance between the B box and the first poly-T terminator, but does not disqualify tRNAs that do not have a terminator signal within 60 nucleotides. This mode is used by default when Cove analysis is not being used as a secondary false positive filter.

Using "strict" parameters with EufindtRNA also incorporates a log odds score for the distance between the B box and the first poly-T terminator, but *rejects* tRNAs that do not have such a signal within 60 nucleotides of the end of the B box. This mode most closely approximates the originally published search algorithm (3); sensitivity is reduced relative to using "relaxed" and "normal" modes, but selectivity is increased which is important if no secondary filter, such as Cove analysis, is being used to remove false positives. This mode will miss most prokaryotic tRNAs since the poly-T terminator signal is a feature specific to eukaryotic tRNAs genes (always use "relaxed" mode for scanning prokaryotic sequences for tRNAs).

-r *file* : save tRNAscan/EufindtRNA formatted output results in *file*

Saves tabular, formatted output results from tRNAscan and/or EufindtRNA first pass scans in *file*. The format is similar to the final tabular output format, except no Cove score is available at this point in the search (if EufindtRNA has detected the tRNA, the negative log likelihood score is given). Also, the sequence ID number and source sequence length appear in the columns where intron bounds are shown in final output. This option may be useful for examining false positive tRNAs predicted by first-pass scans that have been filtered out by Cove analysis.

-u *file* : search with Cove only those sequences & regions delimited in *file* (tabular results file format)

This option allows the user to re-generate results from regions identified to have tRNAs by a previous tRNAscan-SE run. Either a regular tabular result file, or output saved with the -r option may be used as the specified *file*. This option is particularly useful for generating either secondary structure output (-f option) or ACeDB output (-a option) without having to re-scan entire sequences. Alternatively, if the -r option is used to generate the previous results file, tRNAscan-SE will pick up at the stage of Cove-confirmation of tRNAs and output final tRNA predicitons as with a normal run.

Note: the -n and -s options will not work in conjunction with this option. Also, if consecutive sequences have identical names in the sequence file being scanned, only the first sequence will be scanned in the regions defined in the -u *file*.

-F *file* : save first-pass candidate tRNAs in *file* that were then found to be false positives by Cove analysis

This option saves candidate tRNAs found by either tRNAscan and/or EufindtRNA that were then rejected by Cove analysis as being false positives. tRNAs are saved in the FASTA sequence format.

-M *file* : save all sequences without at least one tRNA hit in *file*

This option may be used when scanning a collection of known tRNA sequences to identify possible false negatives (incorreclty missed by tRNAscan-SE) or sequences incorrectly annotated as tRNAs (correctly passed over by tRNAscan-SE). Examination of primary & secondary structure covariance model scores (-H option), and visual inspection of secondary structures (use -F option) may be helpful resolving identification conflicts.

6 Examples

Several C. elegans cosmids and a subset of the Sprinzl tRNA database (animal cytoplasmic + eubacterial) are used in these examples to illustrate various features of the program; all have been included in the /Demo subdirectory so the user may also try out these examples. These files are written in the FASTA sequence format and have the file extension '.fa'. Run times are given for an R4400-200 Indigo SGI, and are approx. equal on a DEC Alpha 2100/400 190MHZ.

⇒ To get a list of run options, type the program name without any parameters or input sequence files

```
> tRNAscan-SE
```

⇒ Default run mode, one sequence & no parameters:

```
> tRNAscan-SE F22B7.fa
```

The following (selected) run options will be printed first:

```
------------------------------------------------------------
Sequence file(s) to search:  F22B7.fa
Results written to:          Standard output
Output format:               Tabular
Searching with:              tRNAscan + EufindtRNA -> Cove
tRNAscan parameters:         Strict
EufindtRNA parameters:       Relaxed (Int Cutoff= -32.1)
------------------------------------------------------------
```

This search produces tabular output results of 5 tRNAs for the 40kbp cosmid F22B7. Takes about 15 seconds.

⇒ Saving output and run statistics files

```
> tRNAscan-SE -o mytrnas -m mystats C28G1.fa
```

The following (selected) run options will be printed first:

```
------------------------------------------------------------
Sequence file(s) to search:  C28G1.fa
Results written to:          mytrnas
Output format:               Tabular
Searching with:              tRNAscan + EufindtRNA -> Cove
```

```
tRNAscan parameters:         Strict
EufindtRNA parameters:       Relaxed (Int Cutoff= -32.1)
Search statistics saved in:  mystats
------------------------------------------------------------
```

Default search mode, saves tabular output results for cosmid C28G1 in file 'mytrnas' and run statistics in file 'mystats'.

⇒ Using '#' as shorthand for default output file names, saving output in ACeDB format

```
 > tRNAscan-SE -a -o# -m# C28G1.fa

------------------------------------------------------------
Sequence file(s) to search:  C28G1.fa
Results written to:          C28G1.out
Output format:               ACeDB
Searching with:              tRNAscan + EufindtRNA -> Cove
tRNAscan parameters:         Strict
EufindtRNA parameters:       Relaxed (Int Cutoff= -32.1)
Search statistics saved in:  C28G1.stats
------------------------------------------------------------
```

This is the same search as the immediately preceding example, but final results are saved in the file 'C28G1.out' and the statistics summary is saved in 'C28G1.stats' using the '#' shorthand for default output file names. Also, the results written to 'C28G1.out' are in the ACeDB format because the -a option was used.

⇒ Saving secondary structure information
Changing name of default output file name prefix

```
 > tRNAscan-SE -p mycosmid -f# -m# C28G1.fa

------------------------------------------------------------
Sequence file(s) to search:  C28G1.fa
Results written to:          Standard output
Output format:               Tabular
Searching with:              tRNAscan + EufindtRNA -> Cove
tRNAscan parameters:         Strict
EufindtRNA parameters:       Relaxed (Int Cutoff= -32.1)
tRNA secondary structure
    predictions saved to:    mycosmid.ss
Search statistics saved in:  mycosmid.stats
------------------------------------------------------------
```

This search sends the tabular results to standard output since no -o option was used. Secondary structure information for all tRNAs was saved in file 'mycosmid.ss' and run stats were saved in 'mycosmid.stats' since the default file name prefix was specified as 'mycosmid' using the -p option.

⇒ Searching only sequences matching a specified name
Searching with Prokaryotic search parameters
Viewing progress of the search analysis

```
 > tRNAscan-SE -d -P -o# -n 'DE*' Sprz-sub.fa

------------------------------------------------------------
Sequence file(s) to search:  Sprz-sub.fa
Search only names matching:  DE*
Results written to:          Sprz-sub.out
Output format:               Tabular
Searching with:              tRNAscan + EufindtRNA -> Cove
tRNAscan parameters:         Strict
EufindtRNA parameters:       Relaxed (Int Cutoff= -36)
------------------------------------------------------------
```

In this example, the program's progress is displayed (-d option) as it searches through the input sequence file in this example. This search will only analyze the 39 sequences in the file Sprz-sub.fa (1013 total seqs) with names matching the 'DE*' key (ie. DE1140, DE1180, DE1200, etc).

Since many of the sequences in this search are from prokaryotes, the -P parameter is used to increase search sensitivity to detect prokaryotic tRNAs that match the consensus A and B boxes less closely than eukaryotic tRNAs (the EufindtRNA intermediate Cutoff is set to -36 instead of the default -32.1; the lower the cutoff, the more sensitive the search). Also, the prokaryotic tRNA model is used in covariance model analysis.

⇒ Search for tRNAs using Cove analysis only

```
 > tRNAscan-SE -C -p F22B7cov -o# -m# F22B7.fa

------------------------------------------------------------
Sequence file(s) to search:  F22B7.fa
Results written to:          F22B7cov.out
Output format:               Tabular
Searching with:              Cove only
Search statistics saved in:  F22B7cov.stats
------------------------------------------------------------
```

For users with the computational resources to spare, sequences can be analyzed using Cove analysis only (tRNAscan and EufindtRNA are not used as a pre-filter). This option is up to 3,000 times slower than using the default search mode, but may detect up to 1% of the tRNAs missed by tRNAscan-SE in default run mode. This example searches the F22B7 cosmid and finds no additional tRNAs over default tRNAscan-SE run mode. Takes about seventy minutes.

⇒ Search using tRNAscan analysis only

```
 > tRNAscan-SE -T -p sprinzl.tscan -l# -m# Sprz-sub.fa

------------------------------------------------------------
Sequence file(s) to search:  Sprz-sub.fa
Results written to:          Standard output
Output format:               Tabular
Searching with:              tRNAscan only
tRNAscan parameters:         Strict
Search log saved in:         sprinzl.tscan.log
Search statistics saved in:  sprinzl.tscan.stats
------------------------------------------------------------
```

Using the -T option approximates using tRNAscan v1.3 only (less sensitive and less selective). The -l option saves a log of the program run in 'sprinzl.tscan.log'. The search finds 962/1013 tRNAs in about 40 seconds, versus the same search in default mode which finds 1007/1013 tRNAs in about 17 minutes (search speed goes down for default mode as density of tRNAs in input sequence(s) goes up).

⇒ Using a previous result file to get secondary structure and ACeDB output without re-scanning entire sequence(s)

First, re-run an example from above, this time saving the tabular output results:

```
 > tRNAscan-SE -o# F22B7.fa
```

Now, suppose you'd like to see the secondary structures (and/or produce ACeDB output) for the tRNAs in this sequence. Instead of re-scanning the entire sequence with a completely new run, specify that tRNAscan-SE should only scan those regions known to have tRNAs from a previous tRNAscan-SE run. In the following example, the pre-existing result file "F22B7.out" is specified with the -u parameter, producing results faster than a complete re-scan of the whole cosmid:

```
 > tRNAscan-SE -u F22B7.out -f# -a -o F22B7.ace F22B7.fa

------------------------------------------------------------
Sequence file(s) to search:  F22B7.fa
Results written to:          F22B7.ace
Output format:               ACeDB
Searching with:              Cove only
Using previous
tabular output file:         F22B7.out
tRNA secondary structure
    predictions saved to:    F22B7.ss
------------------------------------------------------------
```

The secondary structures for these tRNAs are saved in "F22B7.ss" and the results are saved in ACeDB format in the file "F22B7.ace". Useful for re-scanning just tRNAs in large sequences or sequence sets.

⇒ Using an alternate genetic code file

```
 > tRNAscan-SE -g gcode.cilnuc DQ6060.fa

------------------------------------------------------------
Sequence file(s) to search:  DQ6060.fa
Results written to:          Standard output
Output format:               Tabular
Searching with:              tRNAscan + EufindtRNA -> Cove
tRNAscan parameters:         Strict
EufindtRNA parameters:       Relaxed (Int Cutoff= -32.1)
Alternate transl code used:  from file gcode.cilnuc
------------------------------------------------------------
```

The tRNA used in this example is from Tetrahymena thermophila, a ciliate protozoan with an alternate translation of the codon TAA (normally a stop codon) to glutamine. By using the alternate translations specified in the included file "gcode.cilnuc", the correct tRNA type is output.

7 Support / Bug Reports / Requests for added options

This is the first release of tRNAscan-SE, so there are bound to be minor problems that will need to be fixed (although we've made every attempt to catch all problems on SGI, Sun, DEC Alpha, and Intel machines). The user is free to either fix the bug him/herself, or send me detailed information on the problem (email to Todd Lowe, lowe@genetics.wustl.edu), and I will make reasonable efforts to remedy the problem. If you do decide to fix the problem yourself, I would like to be notified so that I may make the change in future updates. Also, if you have suggestions for options that might be added to enhance the usefulness of the program, feel free to suggest them.

8 References

1. Fichant, G.A. and Burks, C. (1991) "Identifying potential tRNA genes in genomic DNA sequences", *J. Mol. Biol.*, **220**, 659-671.

2. Eddy, S.R. and Durbin, R. (1994) "RNA sequence analysis using covariance models", *Nucl. Acids Res.*, **22**, 2079-2088.

3. Pavesi, A., Conterio, F., Bolchi, A., Dieci, G., Ottonello, S. (1994) "Identification of new eukaryotic tRNA genes in genomic DNA databases by a multistep weight matrix analysis of transcriptional control regions", *Nucl. Acids Res.*, **22**, 1247-1256.

4. Lowe, T.M. & Eddy, S.R. (1997) "tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence", *Nucl. Acids Res.*, **25**, 955-964.

A FASTA Sequence Format

The FASTA sequence format consists of a sequence name and description on a single line starting with the greater than symbol '>', followed by the sequence:

```
> SequenceName description here
ATGTCGTTACCGTCGTCGGGACCGACCATG
AGAGCGA
```

More than one sequence can be included in the same file:

```
> Randseq1 first randomly generated seq
GGTGGTTACTAACCGTAAGAGATGATGTCGCCGTGGTCGCGTGGCGCCGCGGACCCAGAT
TGTACTTCTCTGAGTCGTTCTAGATCGACCAGTCTTCTAGCTTGCCCGTGAGGTATGGGG
AGCCGCATATTGCCCACAAT
> Randseq2 second randomly generated seq
GCGACGCGTCTCTACACCAGACGCTTCTGTTGAGGAAGAGTGCCTGAGTGCAGGTCCTCG
AGAACCCACTGGAACTTGAAGGGCGCGTCTCACTGGTCGTGAGAAGGCTCCGTCGATACG
AAAGTCCATGCCAAGGACAT
> Randseq3 third randomly generated seq
GGCGAGTCTGAACTCACAAATATTGCACGAGAGTTTAGTGTATGTTCCTCTTAGGCTGAT
AACAATAGTTTAGTGAGCGGAAATGCAACCGCGAGGCGGTCCCCTGCGCTTGTAATGGCC
ACCTGTTGCCCGTCGGATAT
```

B Distribution and copying terms

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software–to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   (a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

   (b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

   (c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the

Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

   (a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   (b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   (c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

   The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under

this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

   If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

   It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

   This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS